A Proposed Plan Execution Architecture for Advanced Life Support System Control

G. Biswas[1], P. Bonasso[2], S. Abdelwahed[1], E.J. Manders[1], D. Kortenkamp[2], J. Wu[1], and S. Bell[2]

[1]Dept. of EECS & ISIS, Vanderbilt University, Nashville, TN;

[2]NASA Johnson Space Center/ER2, Houston TX.

Advanced life support (ALS) systems require complex control strategies that can maintain stable system performance and balanced resources with small margins and minimal buffers. In closed-loop life support systems there are complex interactions between sub-systems such as air, water, food production, solids processing, and the crew. Recent research at NASA Johnson Space Center has led to significant insights into autonomous control of ALS systems [Leon *et al* 1997; Kortenkamp *et al* 2001; Schreckenghost *et al* 2002]. Routine control of an ALS system is well within the reach of current techniques. For example, the autonomous control system described in [Schreckenghost *et al* 1998] operated around the clock for 73 straight days during a 90 day crewed test with minimal human intervention and the autonomous control system for a recent test of an advanced water recovery system operated with minimal human intervention for over eighteen months[Bonasso *et al* 2002]. However, these control systems are not able to deal convincingly with the concurrent and interacting control of several subsystems, to coordinate the effective and efficient long term management of resources with the planning of mission activities, and to demonstrate effective recovery from significant anomalies. A solution to these issues is needed in order to demonstrate life support systems amenable to efficient long-duration missions such as the human exploration of Mars.

In this paper, we present a proposed multi-level computational architecture that integrates planning and hierarchical control schemes to develop a dynamic planning and control system that is reactive and fault-adaptive, but at the same time, is designed to manage resources for the duration of a long mission. The computational architecture adopts a novel approach to integrating components of the 3T control architecture developed at NASA JSC and Metrica [Bonasso *et al* 1997] with the hierarchical model-based multi-level control systems that have been developed at Vanderbilt University [Abdelwahed *et al* 2005]. Neither 3T nor the multi-level model-based control architecture alone present the complete solution for long-duration autonomous operations. The former lacks the dynamic models necessary to make efficient coordinated use of scarce resources and to maintain smooth transitions among controller states at finer granularity time scales, while the latter lacks domain procedural knowledge to understand the relations between mission goals and planned activities, and to allow the execution of specialized activities, such as maintenance and fault-recovery. Further, it is much harder to provide meaningful interfaces to the user through the control systems.

This integrated computational architecture combines the best of these two approaches. Our general design uses the dynamic models of model-based control architecture to inform the state-based procedural schemas during plan development and execution, as well as to carry out the dynamic control of the habitat subsystems. The 3T planner will provide overarching mission plans, while the 3T sequencer can instantiate procedures that would significantly increase the computational complexity associated with system analysis and decision making with the model-based control architecture.

The 3T planning module drives the supervisory control scheme. Given a top-level goal, such as "conduct habitat operations while supporting extravehicular activities (EVA)", the planner automatically generates a habitat plan for a given duration. The planner reasons in depth about goals, resources and sequencing constraints. It integrates mission goals with *a priori* knowledge, such as the crew schedule, EVA schedule, crop plantings and harvesting, and resource constraints. This knowledge is stored in the world model. During plan generation, the 3T planner draws from the task-resource consumption model of the Resource Manager (middle level), to take into account the dynamic effects of planning decisions. The resulting plan steps and ordering will be tailored to make the best use of scarce resources. Using the user interface capabilities of the

planner, the plan can be reviewed by mission control operators and the habitat crew before going into effect.

The middle level of our combined architecture consists of the 3T sequencer working in concert with the model-based supervisory controller. To execute the plan, the planner passes the next step in the plan for each area of the habitat to the 3T sequencer, which decomposes the plan step into RAPs that are further decomposed until the final sequences are at the level of the system controllers in the third level of the architecture, e.g., the Water Recovery System (WRS) or the crop chambers. An example sequence for the Air Revitalization System (ARS) was given in the previous section. A sequence to sustain crop growth might be 1) harvest a wheat crop, 2) harvest a soybean crop, 3) plant a soybean crop, 4) and harvest a salad crop. The selection of RAPs from the RAP library will be guided by dynamic constraints provided by the models in the model based supervisor also in the middle layer. The resulting sequences are then passed to the supervisory controller through the model information interface, which uses them as ordering constraints; e.g., the supervisor may force the ordering of a set of parallel tasks to ensure that required resources will be produced while not violating energy constraints, or it may adjust the duration of one of the steps as in the previous scenario. Using resource constraints, the supervisory controller transforms the sequence into a schedule of control specifications for the system level controllers, which then carry out the execution sequence for their respective systems (e.g., Air Revitalization (ARS) and Water Recovery (WRS)). Mission controllers and the crew have access to the state of the executing procedures via the system state information access module. This is especially needed when the crew carries out maintenance and ad hoc procedures that do not follow nominal operating schemes.

The system level controllers see each system as an input-output module, where material and energy are input to the system with the goal of producing desired states within the system and output that can be expressed in terms of material, energy, and performance quality parameters. The input-output mappings created by these controllers define utility-based multi-criterion objective functions that the lowest-level subsystem controllers employ to optimize dynamic behavior of subsystems in a way that they minimize the use of resources, while producing the necessary output. For example, given the levels of gases and the amount of energy available to the ARS during the above example sequence period, the system controller for the ARS will regulate the CO_2 and O_2 stores to maximize the CO_2 consumption to support the incineration operations.

Results of the execution from the system controllers are aggregated from the subsystem controllers in the bottom later and provided to the supervisor. In our current architecture, the subsystem controllers are designed to maintain set point control, i.e., maintain the operating region of their respective subsystems at levels and operating modes specified by the system controllers. The supervisor will update its dynamic models as well as pass the execution results to the sequencer as a set of execution states. The RAPS interpreter has the capability to determine new task sequences when faults occur in the system or in the face of unsuccessful execution of task steps. As RAPs sequences complete, the interpreter informs the planner which will update the plan and pass down the next plan step to be executed. Such an update may simply change start and stop times of steps while maintaining the original ordering. If the RAPs interpreter reports a failure of a plan step, as in the case of the faulty CDRA above, the planner may replan the mission steps, adding or omitting steps depending on the effect of the failed step on the overall mission objectives. As in plan generation, the task resource models of the supervisory controller will inform the replanning. As well, users will be able to modify the plan at their discretion as the crew did in the above scenario by requiring that the EVA take place as originally scheduled.

The principle of "cognizant failure" is still embodied in each level of the architecture. The system controllers provide robust regulation of the habitat subsystems, notifying the middle layer of any failing processes. The supervisory controller dynamically adjusts control schedules as the situation changes, informing the sequencer as to the state of tasks. The sequencer in turn serves as the mechanism to invoke alternate procedures as well as fault recovery procedures. Equally important, in light of severe failure, the sequencer will invoke "safing" procedures for the habitat subsystems, informing the planner which in turn will carry out replanning.

Additionally, the user has access to the levels of control where the aggregate of information and control stratagems is meaningful, and yet the complex details of such things as multi-criterion objectives functions remain hidden.

Scenario

We illustrate our proposed architecture through an example scenario. We begin with the assumption of a ninety-day mission plan that is scheduled in 28-day segments. Within the first 28-day period, the mission goal for the habitat might be "to conduct habitat operations while supporting an extravehicular activity (EVA) on day eighteen". An automated planning capability produces a plan of operation that includes tasks to maintain and operate the habitat, operate the water recovery system (WRS), air revitalization system (ARS) and crew quarters climate control, support

the required EVA, sustain crop growth, and ensure safe disposal of solid waste. Using resource models of the dynamics of the habitat subsystems the plan will make efficient use of power, air and water stores and habitat inventories.

Next, a reactive planning capability selects routine procedures for carrying out the first step of each part of the plan for each subsystem. For example, for the ARS:

1) Seven days of nominal operations.
2) Four days in high CO_2 consumption state to clear CO_2 reservoirs in preparation for incineration operations,
3) Four days in an extreme high CO_2 state to scrub the CO_2 resulting from incineration,
4) One day providing O_2 to tanks to be used for the upcoming 24 hour EVA on day eighteen, and
5) Resume nominal operations on day ten.

This sequence is then passed to a dynamic control execution capability that examines the existing resources for the ARS and suggests an extra day to ensure the O_2 tank level increases above a pre-determined value (say 10 kg). Since the extra day will still support the EVA on day eleven, the reactive planner makes no further changes to the ARS execution plan. The dynamic control executive issues time-ordered control specifications for all the habitat systems (WRS, ARS, Power generation, Biomass, etc.) and their corresponding subsystems commensurate with the procedures (i.e., partial plan sequences) from the reactive planner. The subsystem controllers execute the directives "optimally" taking into account the continuous dynamics of the respective subsystem for the first nine days. For example, a change detection algorithm might notice an increase in power usage in the CO_2 removal system (CDRA), but its subsystem controller is able to compensate the increase by decreasing the heater temperature a little, and also adjusting blower and pump speeds.

On day ten, however, the dynamic control executive determines that the CDRA behavior has continued to drift away from the nominal, and the system is operating sub-optimally. By now, the fault detection module has reliably established that there is a restriction in the CO_2 output line and also a leak is detected in the desiccant bed. The system controller has adjusted for this by reducing Oxygen Generation Assembly (OGA) and CO_2 Reduction System (CRS) (Sabatier) operating times, but if this trend continues, air quality in the crew chamber will start dropping below acceptable levels, or lot more energy will have to be directed toward the CDRA. With the night period approaching, this is not considered a good option (by the supervisory control predictor). This situation is reported by the supervisory controller to the RAPS (reactive planner) unit. This unit (the Sequencer) is told that it will now take five days to clear the CO_2 reservoirs. The reactive planner can make no adjustment that will compensate for the extra day and informs the planner. The planner sees the situation and determines there are options at this time such as (i) perform a CDRA repair and, (ii) drop the scheduled EVA activity.

The habitat planner considers the situation, and through its own analysis using its world model determines that a new plan that includes a two-day crew task for repair of the CDRA, which will create an O_2-restricted situation for a few days. As a result, the EVA activity is pushed back to day twenty, since one of the crew repairing the CDRA is also needed for the EVA. Furthermore, the astronauts are required to be cautious while exercising, e.g., none of the crew should exercise at the same time.

At this stage, using an interface to the planner, the habitat commander informs the planner that the EVA task cannot be slipped because it involves a communications experiment that depends on the relative orbits of the moon and the earth about the sun, a constraint unknown to the habitat planner. The planner, in further conference with the model-based resource manager, determines that if the crew completely omits their exercise period until after the EVA, the ARS can meet the incinerator and EVA requirements. The resulting habitat plan omits crew exercise from the crew plan and schedules the CDRA repair after the EVA.

When the CDRA repair takes place, the reactive planner will select an appropriate repair procedure for the crew and a set of modes for ARS and other affected subsystems, and the dynamic controller will execute these changes efficiently. For example, oxygen generation may be suspended, thus reducing the water requirement from the WRS during the repair period. As well, during the repair, the reactive planner will serve as the subsystem level interface to the dynamic controller.

When the repair is complete, the dynamic controller will verify the normal operation of the CDRA and inform the reactive planner, which in turn informs the habitat planner. The habitat planner will adjust the inventory of materials used in the repair and replan if necessary.

A key observation from this scenario is that once anomalous situations are detected, mechanisms kick in at different levels to attempt to contain and compensate for the fault, without having to sacrifice mission goals. For less critical faults of small magnitude, the subsystem controllers can compensate for the change in behavior. At the next level, the system controller may redistribute resources or, if possible reassign some tasks, to keep the system performance and output at different levels. Then the supervisory controller jumps in to determine if it can impose non-critical restrictions to avoid over draining of

resources or reduction in effort without significant loss of capabilities. If the problems persist, the reactive planner or the replanner may be invoked to determine new plans. Last, mission control or the crew may want to change some of the mission goals to avoid potential problems. In all of these situations, decisions made at the top take precedence, which imply that the lower level units, especially the lower-level controllers have to change their strategy to satisfy the new requirements.

References

Abdelwahed, S., J. Wu, G. Biswas, J. Ramirez and E. J. Manders, "Online Fault Adaptive Control for Efficient Resource Management in Advanced Life Support Systems," *Habitation: International Journal for Human Support Research*, Vol. 10, No. 2, pp. 105-116, 2005.

Bonasso, R.P., R. J. Firby, E. Gat, D. Kortenkamp, D. Miller and M. Slack, "Experiences with an Architecture for Intelligent, Reactive Agents," *Journal of Experimental and Theoretical Artificial Intelligence*, Vol. 9, No. 1, 1997.

Bonasso, R. P., David Kortenkamp and Carroll Thronesbery, Intelligent Control of a Water Recovery System. In *AI Magazine*, Vol. 24, No. 1, Spring 2003.

Kortenkamp, D., R. Peter Bonasso and Devika Subramanian, "Distributed, Autonomous Control of Space Habitats," *IEEE Aerospace Conference*, 2001.

Leon, J., David Kortenkamp and Debra Schreckenghost, "A Planning, Scheduling and Control Architecture for Advanced Life Support Systems," *Proceedings of the NASA Workshop on Planning and Scheduling in Space*, 1997.

Schreckenghost, Debra, Mary Beth Edeen, R. Peter Bonasso, and Jon Erickson, "Intelligent Control of the Product Gas Transfer for Air Revitalization," *Proceedings of the 28th Conference on Environmental Systems*, 1998.

Schreckenghost, Debra, Carroll Thronesbery, R. Peter Bonasso, David Kortenkamp and Cheryl Martin, "Intelligent Control of Life Support for Space Missions," in IEEE *Intelligent Systems Magazine*, Vol. 17, No. 5, September/October 2002.